پروژه اول علوم اعصاب محاسباتی

- صورت پروژه در این آدرس قابل مشاهده است.
- با توجه به دشواری حفظ ساختار صورت پروژه در گزارش، آن ساختار نادیده گرفته شده و مطالب با ساختاری مناسب برای دنبال کردن نمودارها و مطالب منظم شده‌اند؛ با اینحال تمام مطالب خواسته شده در صورت پروژه، در این گزارش پوشانده شده‌اند.

0. فهرست مطالب

1. مدل LIF

برای پیاده‌سازی این مدل نورونی، کافی است اختلاف پتانسیل نورون را در هر گام طبق رابطه زیر به روزرسانی کنیم:

$$U(t+\Delta) = U(t) - \frac{\Delta}{\tau}[(U(t) - U_{rest}) - R.I(t)]$$

$$if \;\; U(t) > U_{spike-threshold} : U(t) = 0 \quad and \quad spike - on!$$

با توجه به رابطه‌ی بالا، انتظارات زیر را از رفتار این مدل نورونی داریم که در ادامه صحت آن‌ها را بررسی می‌کنیم:
1- با افزایش میزان جریان ورودی، اختلاف پتانسیل مثبت‌تر و درنتیجه فرکانس spike خروجی بیشتری شاهد هستیم.
2- با افزایش میزان مقاوت (R)، میزان تأثیرپذیری اختلاف پتانسیل نورون از جریان ورودی بیشتر می‌شود. به زبان ساده‌تر، با دریافت جریان ورودی، اختلاف پتانسیل با سرعت بیشتری افزایش پیدا می‌کند (مثبت می‌شود) و درنتیجه فرکانس spike خروجی افزایش پیدا می‌کند.
3- با افزایش τ، به صورت کلی سرعت تغییرات اختلاف پتانسیل کاهش پیدا می‌کند (لختی میزانی اختلاف

پتانسیل بیشتر می‌شود).
4- با کاهش مقدار اختلاف $U_{spike-threshold}$ و U_{rest}، شاهد فرکانس بیشتری در spikeها خواهیم بود.

In [2]:
```
from cnsproject.network.neural_populations import LIFPopulation
from cnsproject.network.monitors import Monitor
from cnsproject.plotting.plotting import Plotter
from cnsproject.utils import *
import matplotlib.pyplot as plt
import torch
```

بجز ابزار شبیه‌سازی (که import شده‌اند)، تابع پایین در این تمرین خاص، برای شبیه‌سازی و مقایسه نورون‌ها در کنار هم به ما کمک خواهد کرد. همچنین در این تمرین، هر شبیه‌سازی را به مدت 250ms ادامه خواهیم داد.

In [3]:
```
def neuron_behaviour(neuron_cls, I, p, time=250, postfix='', name='', **args)
    neuron = neuron_cls((1,), **args)
    monitor = Monitor(neuron, state_variables=["s", "u"], time=time)
    neuron.refractory_and_reset()
    monitor.simulate(neuron.forward, {'I': I})
    p.monitor = monitor
    p.neuron_spike('s'+postfix, x_vis=False, y_label='spikes', title=name)
    p.neuron_voltage('u'+postfix, x_vis=False, y_label="u (mV)", x_label='')
    p.current_dynamic('i'+postfix, I=I, y_label="I (mA)", repeat_till=time)

time=250 #ms
```

2. ورودی جریان تصادفی

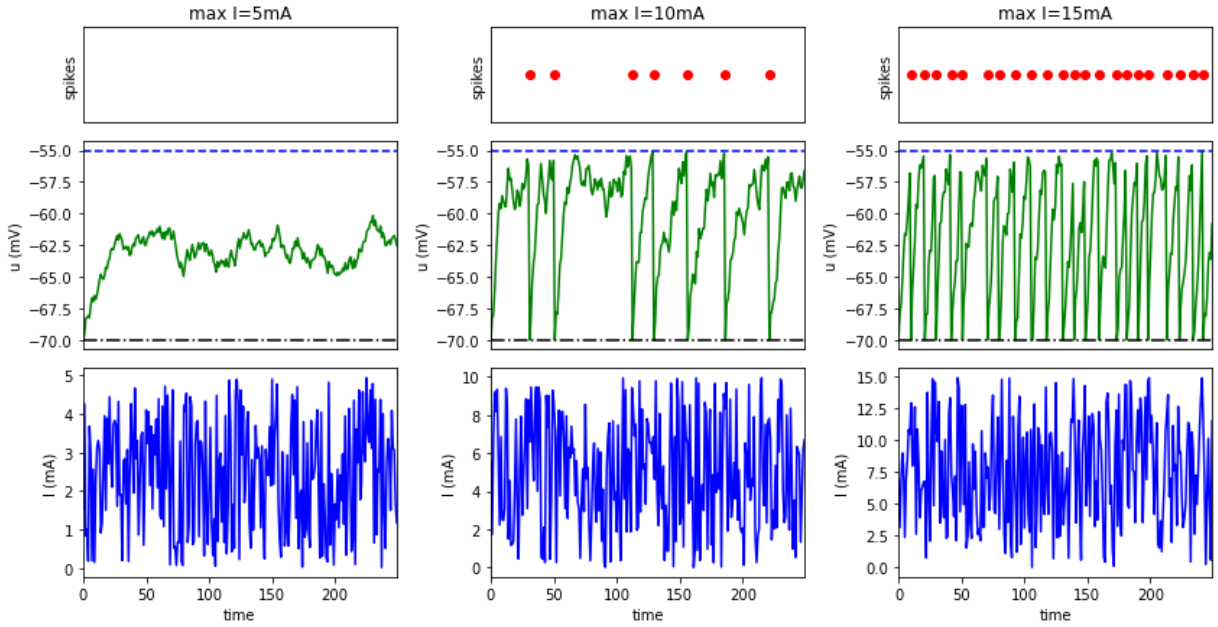
برای بررسی درستی کارکرد مدل، در ابتدا رفتار این مدل در مقابل ورودی‌های تصادفی را مورد بررسی قرار می‌دهیم.

A.2. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

سه نوع جریان ورودی تصادفی با دامنه‌های متفاوت (۵، ۱۰ و ۱۵ میلی‌آمپر) تولید کرده و رفتار مدل در برابر این سه ورودی را در کنار هم رسم می‌کنیم. انتظار داریم با ورودی شدیدتر (دامنه بزرگ‌تر)، مدل فرکانس spike خروجی بیشتری داشته باشد (انتظار شماره ۱ در لیست ابتدای گزارش).

In [4]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*5
neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=10., name

I = torch.rand(time,1)*10
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name

I = torch.rand(time,1)*15
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=10., name
```

```
p.show()
```

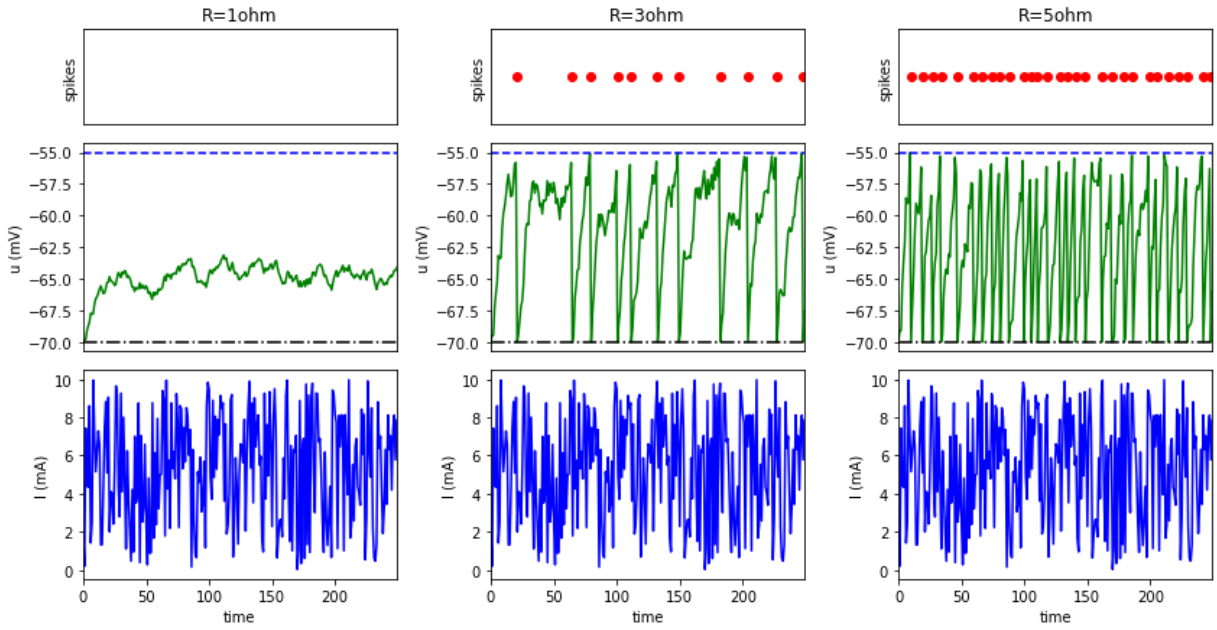


مطابق بند اول از انتظارات ما از مدل، با جریان‌های ورودی شدیدتر، شاهد فرکانس بیشتر در spikeهای خروجی هستیم. جالب از که با جریان ورودی با دامنه 5mV، اختلاف پتانسیل نورون هیچ‌گاه به مرز spike نمی‌رسد!

B.2. بررسی رفتار مدل با مقاوت‌های متفاوت

حال از جریان تصادفی با دامنه ۱۰میلی‌ولت استفاده می‌کنیم تا تأثیر تغییر مقاوت مدل نورونی را مورد بررسی قرار دهیم.

In [5]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*10

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=1., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=5., tau=10., name

p.show()
```

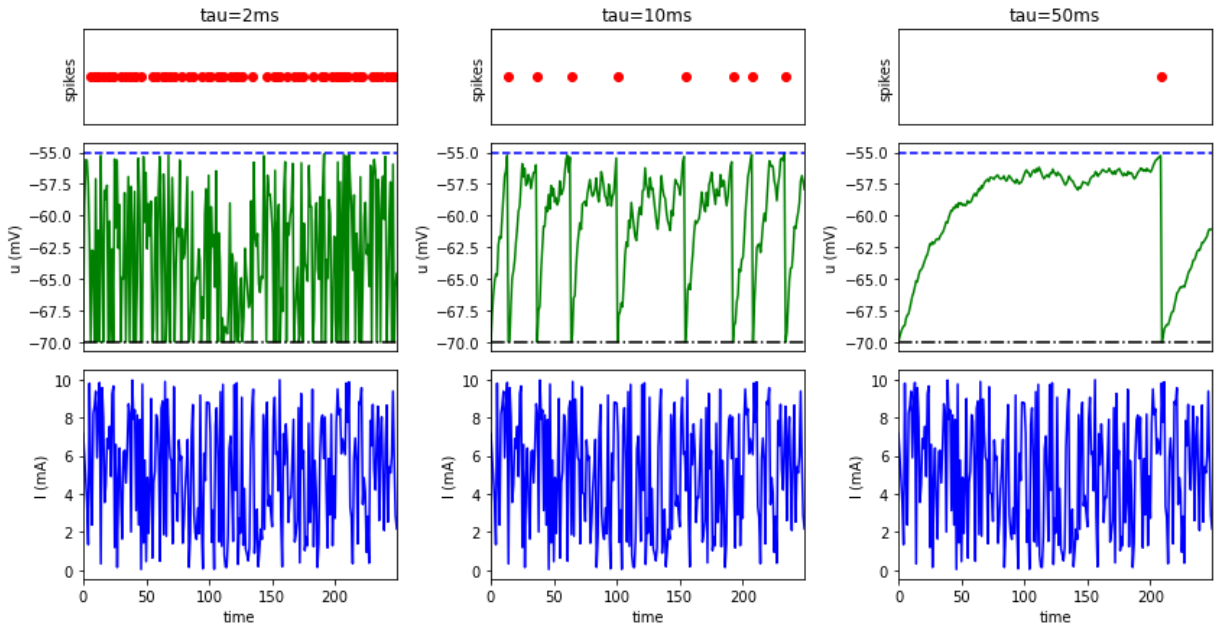


مطابق بند ۲ از انتظارات ما از مدل، شاهد افزایش فرکانس spike خروجی با افزایش مقاوت نورونی هستیم.

C.2. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

حال از جریان تصادفی با دامنه ۱۰میلی‌ولت و مقاومت ۳اهم استفاده می‌کنیم تا تأثیر تغییر ثابت‌زمانی مدل نورونی را مورد بررسی قرار دهیم.

In [6]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*10

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=2., name=
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=50., name

p.show()
```

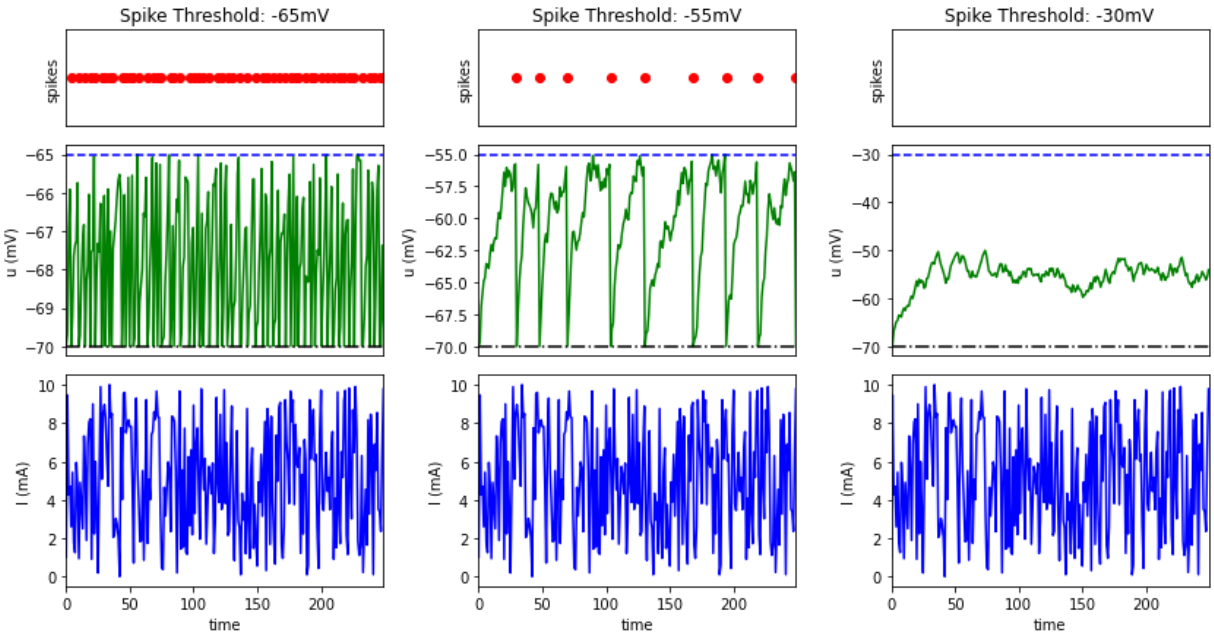


مطابق بند ۳ از انتظارات ما از مدل، شاهد کاهش سرعت تغییرات اختلاف پتانسیل و فرکانس spike خروجی در اثر افزایش مقدار ثابت زمانی مدل نورونی هستیم.

D.2. بررسی رفتار مدل با *spike−threshold*های متفاوت

حال از جریان تصادفی با دامنه ۱۰میلی‌ولت، مقاومت ۳اهم و ثابت زمانی ۱۰میلی‌ثانیه استفاده می‌کنیم تا تأثیر تغییر *spike−threshold* مدل نورونی را مورد بررسی قرار دهیم.

In [7]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*10

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=10., spik
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., spik
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=10., spik

p.show()
```



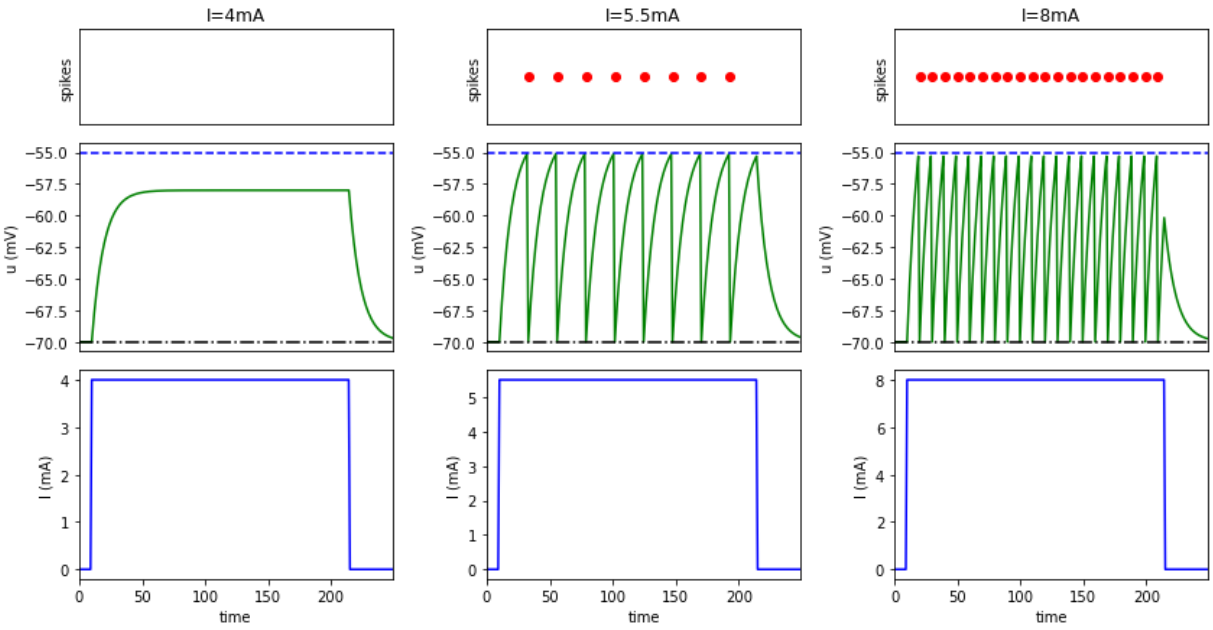
مطابق بند ۴ از انتظارات ما از مدل نورونی، با افزایش اختلاف بین spike-threshold و resting-potential، فرکانس spike خروجی کاهش پیدا می‌کند (چون رسیدن اختلاف پتانسیل نورون به مرز spike دشوارتر می‌شود.

3. ورودی جریان ثابت

پیشتر شاهد مطابقت مشاهدات حاصل از شبیه‌سازی با انتظارات خود از مدل بودیم. حال به بررسی آزمایش‌های یکسان اما این بار با ورودی جریان ثابت خواهیم پرداخت. رفتار نورون در برابر جریان ثابت، منظم‌تر خواهد بود.

A.3. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

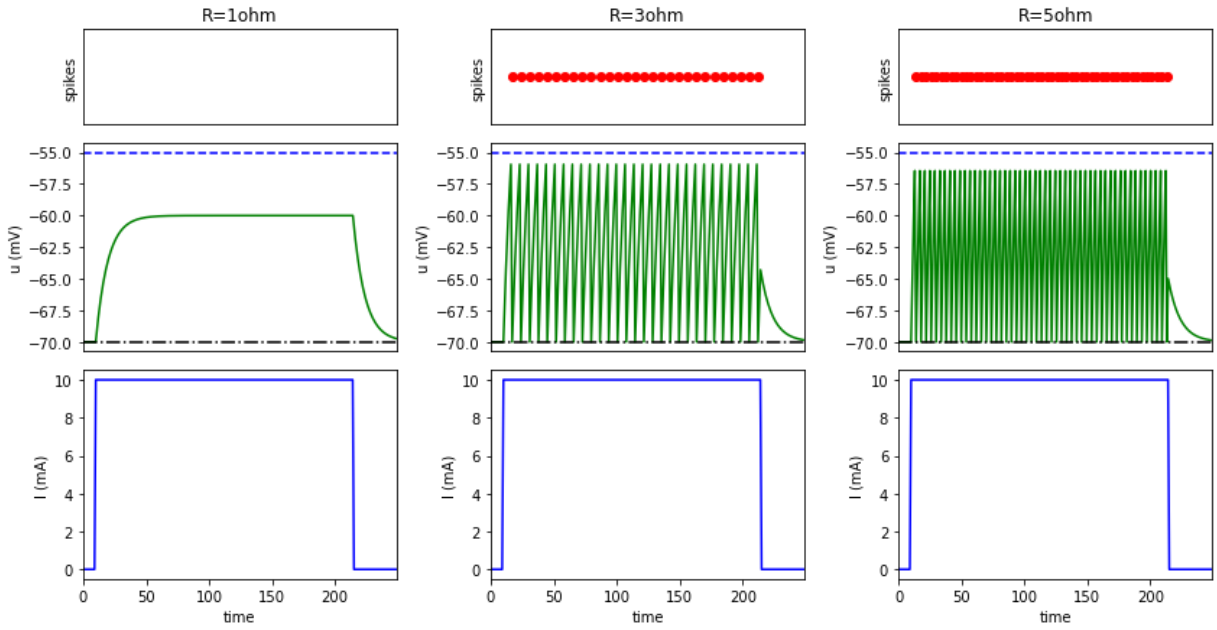
In [8]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

Ion = 4
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)
neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=10., name

Ion = 5.5
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name

Ion = 8
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=10., name

p.show()
```



مطابق بند اول از انتظارات ما از مدل، با جریان‌های ورودی شدیدتر، شاهد فرکانس بیشتر در spikeهای خروجی هستیم. جالب از که با جریان ورودی با دامنه 4mV، اختلاف پتانسیل نورون هیچ‌گاه به مرز spike نمی‌رسد!

B.3. بررسی رفتار مدل با مقاوت‌های متفاوت

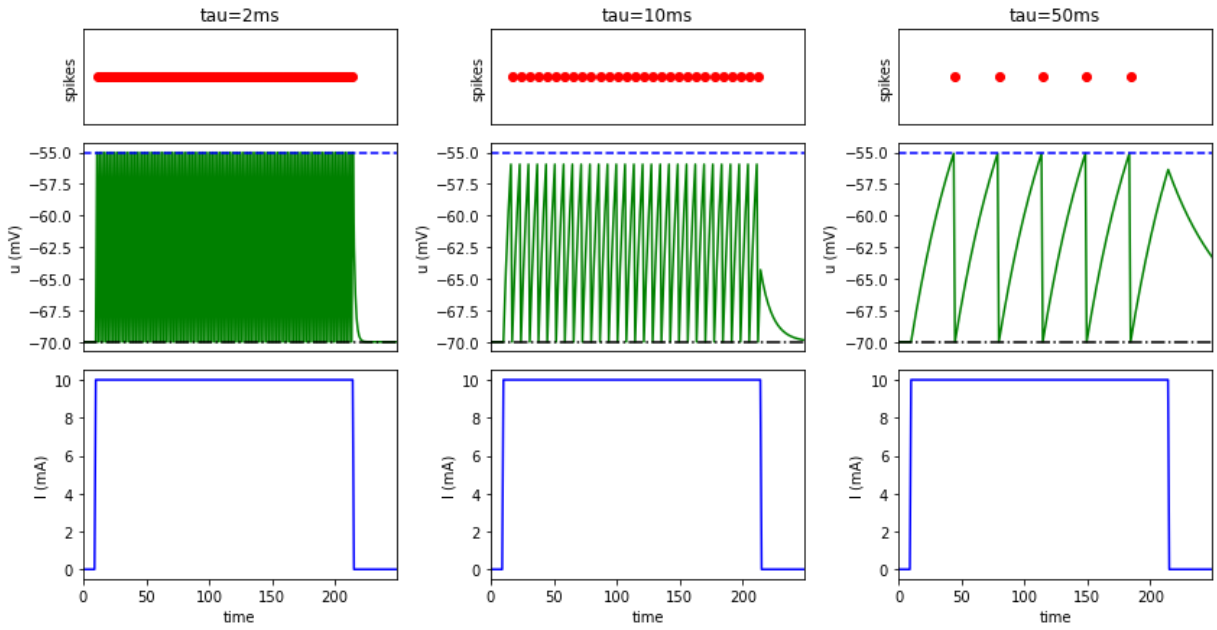
In [9]:
```
Ion = 10
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=1., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=5., tau=10., name

p.show()
```



مطابق بند ۲ از انتظارات ما از مدل، شاهد افزایش فرکانس spike خروجی با افزایش مقاوت نورونی هستیم.

C.3. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

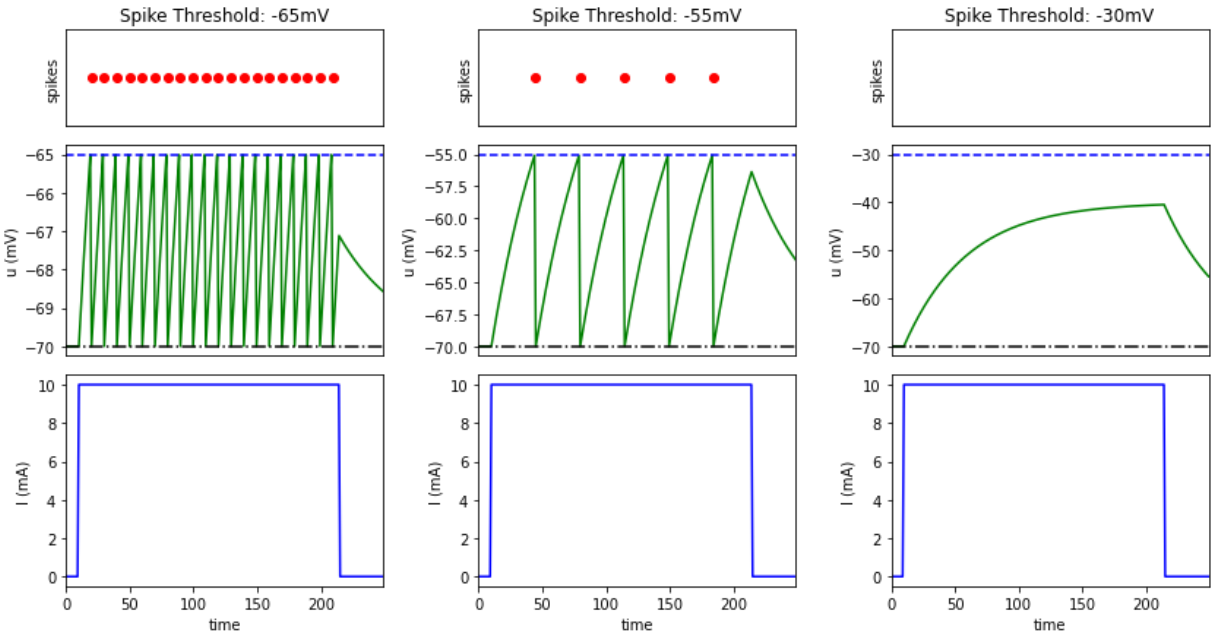
In [10]:
```
Ion = 10
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=2., name=
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=50., name

p.show()
```



مطابق بند ۳ از انتظارات ما از مدل، شاهد کاهش سرعت تغییرات اختلاف پتانسیل و فرکانس spike خروجی در اثر افزایش مقدار ثابت زمانی مدل نورونی هستیم.

D.3. بررسی رفتار مدل با *spike−threshold*های متفاوت

In [11]:
```
Ion = 10
I = step_function(time, 10, val1=Ion) - step_function(time, 215, val1=Ion)

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=50., spik
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=50., spik
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=50., spik

p.show()
```



مطابق بند ۴ از انتظارات ما از مدل نورونی، با افزایش اختلاف بین spike-threshold و resting-potential، فرکانس spike خروجی کاهش پیدا می‌کند (چون رسیدن اختلاف پتانسیل نورون به مرز spike دشوارتر می‌شود.

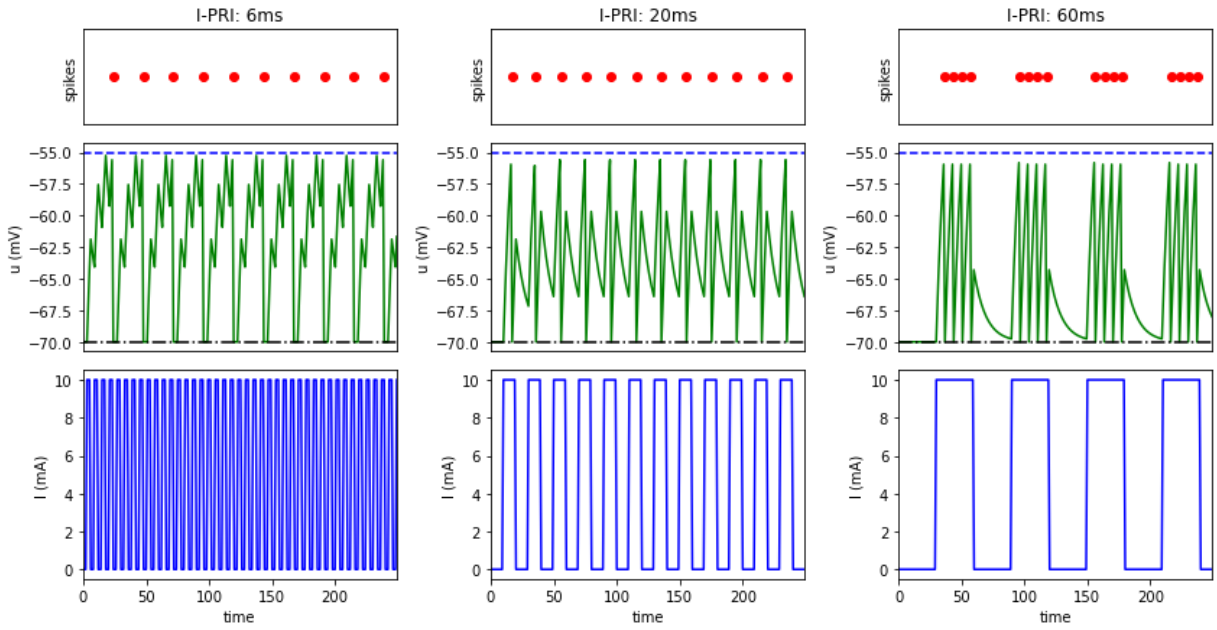
4. ورودی جریان‌های دیگر

در این بخش، از سر کنجکاوی به بررسی رفتار مدل نورونی در برابر چند نوع خاص از جریان‌های ورودی و مجموعه پارامترهای متفاوت خواهیم پرداخت.

A.4. ورودی جریان پالس مربعی متوازن

در این بخش، جریان‌های ورودی به شکل پالس مربعی با دوره‌تناوب‌های متفاوت را بررسی خواهیم کرد.

In [12]:
```
Ion = 10

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = [0]*3+[Ion]*3
neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=10., name

I = [0]*10+[Ion]*10
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name

I = [0]*30+[Ion]*30
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=10., name

p.show()
```

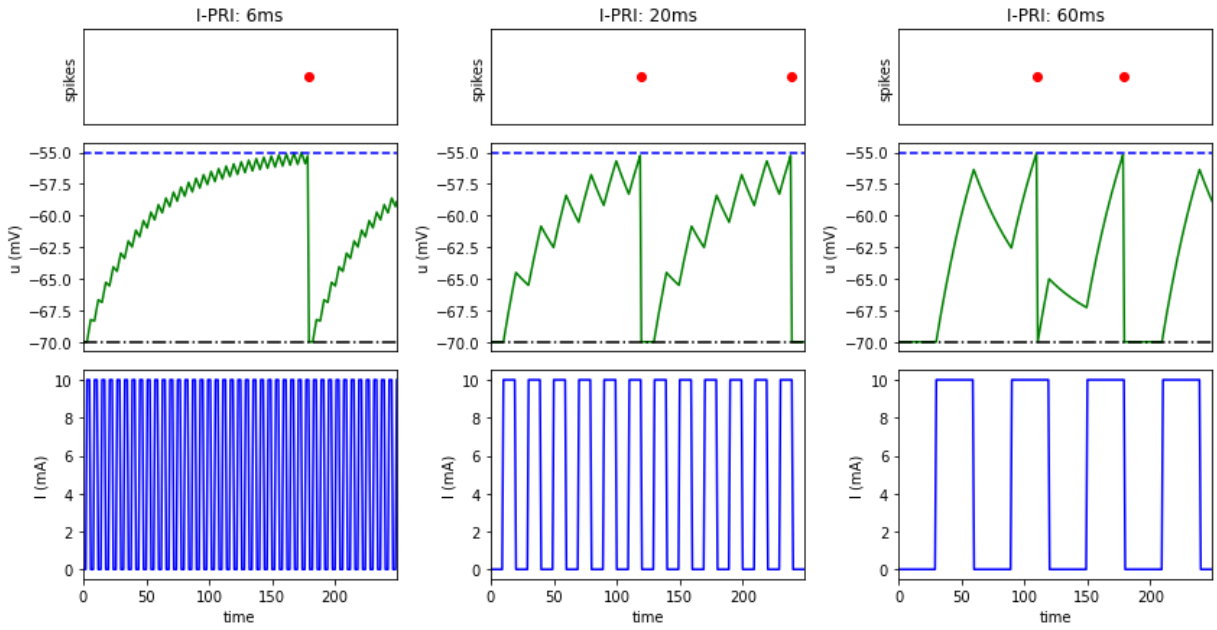


نتیجه مطابق انتظار است. هرچه دوره تناوب بزرگ‌تری داشته باشیم، به نورون مهلت کمتری برای جلوگیری از spike داده می‌شود. همچنین در مقاطعی که پالس خاموش است، اختلاف پتانسیل نورون به حالت rest نزدیک می‌شود.

B.4. ورودی جریان پالس مربعی متوازن با ثابت زمانی مدل نورونی بزرگ

با توجه به تاثیر ثابت زمانی مدل نورونی و رفتار مدل در برابر جریان پالس مربعی، انتظار می‌رود با حفظ این جریان ورودی و افزایش ثابت زمانی مدل، مانع رسیدن اختلاف پتانسیل نورون به مرز spike شویم و در زمان خاموشی پالس ورودی، مدل بتواند اختلاف پتانسیل خود را تخلیه کند و به این ترتیب، فرکانس spike خروجی بشدت کاهش پیدا کند.

In [13]:
```
Ion = 10

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = [0]*3+[Ion]*3
neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=50., name

I = [0]*10+[Ion]*10
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=50., name

I = [0]*30+[Ion]*30
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=50., name

p.show()
```

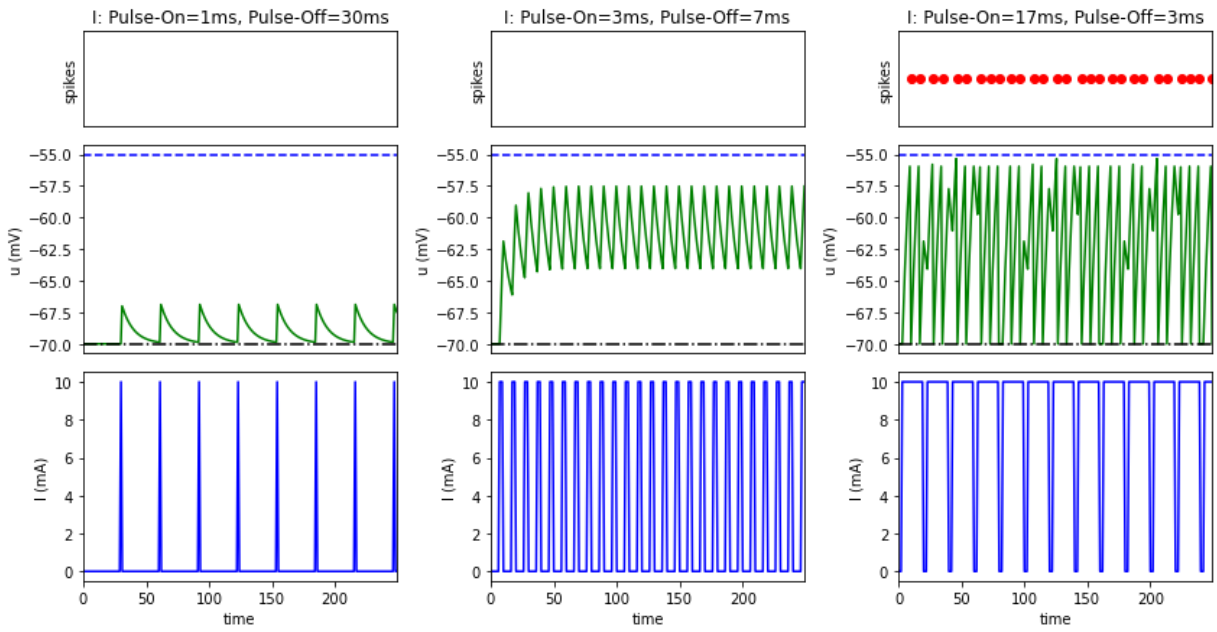


نتیجه مطابق انتظار است.

C.4. ورودی جریان پالس مربعی غیر متوازن

انتظار داریم با افزایش نسبت مدت زمان خاموشی پالس ورودی و به مدت زمان روشن بودن این پالس، مدل فرصت بیشتری برای تخلیه پتانسیل پیدا کند و بدین‌ترتیب، فرکانس spike را کاهش دهد.

In [14]:
```
Ion = 10

plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = [0]*30+[Ion]*1
neuron_behaviour(LIFPopulation, I, p, postfix='1', dt=1., R=3., tau=10., name

I = [0]*7+[Ion]*3
neuron_behaviour(LIFPopulation, I, p, postfix='2', dt=1., R=3., tau=10., name

I = [0]*3+[Ion]*17
neuron_behaviour(LIFPopulation, I, p, postfix='3', dt=1., R=3., tau=10., name

p.show()
```



نتیجه مطابق انتظار است.

5. F-I Curve

در این بخش، رفتار نورون را در برابر میزان شدت جریان ورودی (ثابت)، با استفاده از نمودار F-I مشاهده می‌کنیم.

In [15]:
```
I_range = 20
def cal_FI(neuron, I_range=I_range):
    monitor = Monitor(neuron, state_variables=["s", "u"], time=time)
    f = []
    for i in range(I_range):
        neuron.refractory_and_reset()
        I = torch.ones(time,1)*i
        monitor.simulate(neuron.forward, {'I': I})
        f.append(sum(monitor['s'])/time)
    return f

plt.figure(figsize=(14,8))
p = Plotter([
    [None,'F','F','F','F',None],
    ['R','R','T','T','S','S'],
], hspace=0.3)
colors = ['blue','green','yellow']

neuron = LIFPopulation((1,), dt=1., R=3., tau=10.)
p.plot('F', y='F', x='I', data={'F':cal_FI(neuron,2*I_range), 'I':list(range(
       color='red', y_label="spike frequency (kHz)", x_label="I (mA)", title=
p['F'].legend()

for i,R in enumerate([1,3,5]):
    neuron = LIFPopulation((1,), dt=1., R=R, tau=10.)
    p.plot('R', y='F', x='I', data={'F':cal_FI(neuron), 'I':list(range(I_rang
           label=f"R={R}mA", color=colors[i], y_label="spike frequency (kHz)"
p['R'].legend()

for i,tau in enumerate([2,10,50]):
    neuron = LIFPopulation((1,), dt=1., R=3., tau=tau)
    p.plot('T', y='F', x='I', data={'F':cal_FI(neuron), 'I':list(range(I_rang
```

```
            label=f"tau={tau}ms", color=colors[i], x_label="I (mA)", title="F-
p['T'].legend()

for i,st in enumerate([-65,-55,-30]):
    neuron = LIFPopulation((1,), dt=1., R=3., tau=50., spike_threshold=st)
    p.plot('S', y='F', x='I', data={'F':cal_FI(neuron), 'I':list(range(I_rang
            label=f"ST={st}mV", color=colors[i], y_vis=False)
p['S'].legend()

p.show()
```